**Assignment – 1**

**Random Walk**

**Conclusion:** Assume the drunk man has completed his random walk, which consists of ‘n’ steps, and is now at position (x, y) in the space, which is at a distance ‘d’ from the lamp post. Since ‘d’ is the Euclidean Distance, the formula to calculate that distance from the origin (0, 0) is:

**Euclidean Distance = √ (x2 + y2)**

From the experiment, we observe that the relationship between the number of steps ‘n’ and the distance ‘d’ is directly proportional to each other **(n ∝ d)**. As observed on the graph, the distance increases as the value of steps increases. The concluded approximate relation between ‘n’ and ‘d’ is:

**d ≈ √n**

**Evidence**

* **Graph between Steps (n) and distance (d)**

Graphical user interface, chart

Description automatically generated

* **Spreadsheet**

|  |  |  |
| --- | --- | --- |
| **Steps** | **Distance** | **Experiment** |
| 5 | 1.943686083 | 10 |
| 5 | 2.285259882 | 20 |
| 5 | 1.838219031 | 30 |
| 5 | 2.173452323 | 40 |
| 5 | 2.063689412 | 50 |
| 5 | 2.057960107 | 60 |
| 5 | 2.001860695 | 70 |
| 5 | 1.858684095 | 80 |
| 5 | 2.005404622 | 90 |
| 5 | 2.056768991 | 100 |
| 20 | 4.71679267 | 10 |
| 20 | 3.430579635 | 20 |
| 20 | 3.731951876 | 30 |
| 20 | 4.204159096 | 40 |
| 20 | 3.633268602 | 50 |
| 20 | 4.070333879 | 60 |
| 20 | 4.032864281 | 70 |
| 20 | 3.626502997 | 80 |
| 20 | 3.854142684 | 90 |
| 20 | 4.131078668 | 100 |
| 75 | 8.519822409 | 10 |
| 75 | 8.529553783 | 20 |
| 75 | 9.2374215 | 30 |
| 75 | 7.932839851 | 40 |
| 75 | 8.099552918 | 50 |
| 75 | 7.6394692 | 60 |
| 75 | 8.056147828 | 70 |
| 75 | 7.725106644 | 80 |
| 75 | 7.08363371 | 90 |
| 75 | 6.743569525 | 100 |
| 200 | 11.46370129 | 10 |
| 200 | 12.25723317 | 20 |
| 200 | 13.44497098 | 30 |
| 200 | 13.30337543 | 40 |
| 200 | 11.83403073 | 50 |
| 200 | 13.65338885 | 60 |
| 200 | 14.33786955 | 70 |
| 200 | 12.28538628 | 80 |
| 200 | 12.01956453 | 90 |
| 200 | 12.76992977 | 100 |
| 350 | 12.07959514 | 10 |
| 350 | 15.58818925 | 20 |
| 350 | 16.42343665 | 30 |
| 350 | 17.73988465 | 40 |
| 350 | 17.15376806 | 50 |
| 350 | 15.61144898 | 60 |
| 350 | 17.62627925 | 70 |
| 350 | 18.47587511 | 80 |
| 350 | 15.95545479 | 90 |
| 350 | 16.00945272 | 100 |
| 500 | 16.80087889 | 10 |
| 500 | 18.68241876 | 20 |
| 500 | 21.07339852 | 30 |
| 500 | 18.93513186 | 40 |
| 500 | 17.58538011 | 50 |
| 500 | 17.02965829 | 60 |
| 500 | 22.02836835 | 70 |
| 500 | 19.19357028 | 80 |
| 500 | 20.27895523 | 90 |
| 500 | 18.50565349 | 100 |

* **Graph**

**Code:**

/\*

\* Copyright (c) 2017. Phasmid Software

\*/

package edu.neu.coe.info6205.randomwalk;

import java.util.Random;

public class RandomWalk {

private int x = 0;

private int y = 0;

private final Random random = new Random();

/\*\*

\* Private method to move the current position, that's to say the drunkard moves

\* @param dx the distance he moves in the x direction

\* @param dy the distance he moves in the y direction

\*/

private void move(int dx, int dy) {

// TO BE IMPLEMENTED

// System.out.println("dx: " +dx+", dy: "+dy);

this.x += dx;

this.y += + dy;

}

/\*\*

\* Perform a random walk of m steps

\*

\* @param m the number of steps the drunkard takes

\*/

private void randomWalk(int m) {

// TO BE IMPLEMENTED

for (int i = 0; i < m; i++) {

this.randomMove();

}

}

/\*\*

\* Private method to generate a random move according to the rules of the

\* situation. That's to say, moves can be (+-1, 0) or (0, +-1).

\*/

private void randomMove() {

boolean ns = random.nextBoolean();

int step = random.nextBoolean() ? 1 : -1;

move(ns ? step : 0, ns ? 0 : step);

}

/\*\*

\* Method to compute the distance from the origin (the lamp-post where the

\* drunkard starts) to his current position.

\*

\* @return the (Euclidean) distance from the origin to the current position.

\*/

public double distance() {

// TO BE IMPLEMENTED

// System.out.println(" x: "+x+" y: "+y);

return Math.sqrt((x \* x) + (y \* y));

}

/\*\*

\* Perform multiple random walk experiments, returning the mean distance.

\*

\* @param m the number of steps for each experiment

\* @param n the number of experiments to run

\* @return the mean distance

\*/

public static double randomWalkMulti(int m, int n) {

double totalDistance = 0;

for (int i = 0; i < n; i++) {

RandomWalk walk = new RandomWalk();

walk.randomWalk(m);

totalDistance = totalDistance + walk.distance();

}

return totalDistance / n;

}

public static void main(String[] args) {

//if (args.length == 0)

// throw new RuntimeException("Syntax: RandomWalk steps [experiments]");

int m = 500; //Integer.parseInt(args[0]);

//int n = 100;

//if (args.length > 1) n = Integer.parseInt(args[1]);

int[] arr = new int[]{10, 20, 30, 40, 50, 60, 70, 80, 90, 100};

for (int n = 0; n < 10; n++) {

double meanDistance = randomWalkMulti(m, arr[n]);

System.out.println(m + "\t" + meanDistance+"\t"+arr[n]);

}

// System.out.println(m + " steps: " + meanDistance + " over " + n + " experiments");

}

}

**Test Cases:**

Graphical user interface, text, application, Teams

Description automatically generated